Automated Translation at Grenoble University

Bernard Vauquois and Christian Boitet

Groupe d'Etudes pour la Traduction Automatique
Boite Postale 68
Université de Grenoble
38402 Saint-Martin-d'Hères, France

Introduction

The authors thank Dr. Slocum for the opportunity to present the work on machine translation at Grenoble. The plan he has proposed for the contributions to this special issue was certainly a very good starting point, as a common frame to present various systems around the world.

It is, however, inevitable that we could not completely fit into it, so that we have sometimes taken some liberty for which we hope to be excused.

1 Project History

1.1 Dates and Funding Sources

Founded in 1972, GETA is one of the laboratories of the Computer Science Department at Grenoble University. From its inception it was supported by CNRS, the French National Center for Research, by means of association contracts renewed every four years, and by the University, having the status of "University Research Team associated with CNRS". Also, several projects have been partially supported by contracts from the Ministries of Defense, Telecommunications, and Industry.

Before that, in 1961, CNRS had created CETA (Centre d'Etudes pour la Traduction Automatique) as a "laboratoire propre", that is, a laboratory supported by CNRS funds and various contracts but not by the University.

From 1961 to 1971, CETA elaborated some basic software tools for MT, and experimented with it mainly on translation from Russian into French. Some investigations were also carried out into the analysis of German and Japanese. In 1971, a corpus including about 400,000 running words had been translated from Russian into French, after several years of development of the various grammars and dictionaries (the "lingware", so to speak).

The system was typically second generation: finite-state morphological analysis, augmented context-free syntactic analysis, direct mapping into a dependency analysis, procedural semantic analysis (using a specialized low-level programming language to transform tree structures) into an interlingua (the famous "language pivot", or "pivot language"), lexical transfer, syntactic generation (using the same low-level tree-transformation language), and morphological generation (beginning with a recursive top-down procedure applied to the tree to get the correct order of the leaves, to be transformed into words).

Since 1972, another approach has been followed that relies more on the use of SLLPs (Specialized Languages for Linguistic Programming); that is, on procedural techniques like controlled production systems and heuristic programming. Also, a main goal has been to design the basic software and the various lingwares with a view to multilingual translation.

A completely integrated programming environment (ARIANE-78) has been developed and used to build a variety of linguistic models, in order to test the general multilingual design and the various facilities for lingware preparation, debugging, and actual use, as well as for human pre- and post-editing.

1.2 Languages Translated; Project and System Sizes

The first three subsections below describe the kinds of experiments performed at GETA.

1.2.1 Writing of Very Reduced Scale Models

The writing of very reduced scale models is mainly oriented to training researchers in the methodology of MT and the use of the various SLLPs available under ARIANE-78.

As an example, Feng Zhi Wei (ISTIC, Beijing) has written a small multilingual translation system from Chinese into Japanese, French, English, German, and Russian, that (of course) uses the same analysis modules for all language pairs.

0362-613X/85/010028-36$03.00

A transfer from English (using a bigger model of English analysis) to Chinese and a generation of Chinese has been produced by Ping Yang (ISTIC). In a similar way, using the same analysis of English, Professor Jun-Ichi Tsujii (of Kyoto University) conducted an experiment into English-Japanese translation.

1.2.2 CONSTRUCTION OF SMALL SCALE MODELS

The construction of small scale models – that is, with small dictionaries but medium size grammars – has been done for feasibility studies and training.

A French-English system was constructed by J.Ph. Guilbaud and M. Dymetman in the framework of a feasibility study for the Ministry of Telecommunications during 1981. This system has recently been reused to implement a rough "self grading" technique to be used in connection with a translator work station (A. Melby, BYU).

Another example is the development of a German-French system, using the same generation as the Russian-French system (see below). This work was started around 1979, with J. Ph. Guilbaud of GETA in charge of morphological analysis and transfer, and Professor Stahl (Paris) in charge of structural analysis.

In cooperation with IFCI (Institut de Formation et de Conseil en Informatique, Grenoble), another system, BEX-FEX, has been developed for teaching purposes (B. Roudaud, S. Chappuy, and E. Guilbaud). The analysis is a simplified version of the IN1 analysis (see below), and the dictionaries are purposely very small (500 lexical units for each language).

In order to give an idea of the complexity of such a system, rather than measure the number of grammatical rules, because they can be very simple or very complex, we measured the number of source lines of the grammars written in the SLLPs ATEF, ROBRA, or SYGMOR. BEX-FEX contains

420	lines for the morphological	analysis
2500	lines for the structural	analysis
300	lines for the structural	transfer
1000	lines for the syntactic	generation
100	lines for the morphological	generation

Also in connection with IFCI, a feasibility study for an English-Arabic system was started in 1984.

1.2.3 BUILDING OF LARGE SCALE SYSTEMS

Large scale systems have been built at the level of laboratory prototypes.

The largest system with regard to the vocabulary, and the most experimented with, is the Russian-French system RUB-FRB. According to an estimate made in June 1984, the dictionaries contain 7000 Russian lexical units (about 17000 words) and 5600 French lexical units (about 16000 words).

By "words", we mean here **lemma**, that is, a normal form of an occurrence, usually used to identify an item in a "natural" dictionary. The various grammars contain

1350	lines for the morphological	analysis
3100	lines for the structural	analysis
380	lines for the structural	transfer
930	lines for the syntactic	generation
120	lines for the morphological	generation

Since 1983 this system has been used to test an experimental "translation unit" using a production-oriented subset of ARIANE-78, PROTRA. Photocopies of technical abstracts from the *Referativnyij Zhurnal* are regularly received by the unit, manually inputted, checked for typing errors (by using the morphological analyzer), machine translated, manually revised (using a multiwindow editor), and sent back. The revision effort includes the search for the source of errors encountered.

As it is, and taking into account that manual input represents the major bottleneck, around 50 to 60 abstracts per month (5000 to 7500 running words) are processed. Fields covered include space sciences and metallurgy.

Next comes the English-Malay system (IN1-BM1). In 1979, a cooperative project with USM (Universti Sains Malaysia, Penang) was launched. The aim of the project is to produce an English-Malay system for the translation of teaching material in technical fields. The level of laboratory prototype should be attained by the end of 1984, when systematic tests will be performed.

The analysis part has been jointly developed by GETA and USM. It is the same that was mentioned earlier: having initially been started for this project, it has been reused for experiments of translation into other languages. As far as size is concerned, the dictionaries contain 1800 English lexical units (about 3000 words) and 1800 Malay lexical units (about 2700 words). The various grammars contain

420	lines for the morphological	analysis
5000	lines for the structural	analysis
600	lines for the structural	transfer
1500	lines for the syntactic	generation
670	lines for the morphological	generation

More recently, a similar project was started with several Universities of Thailand (Chulalongkorn and Rakhamhaeng in Bangkok, Prince of Songkla in Had-Yai) for translation from English into Thai.

Last but not least, ARIANE-78 and the related linguistic techniques have been selected as the basis for an industrial development, in the framework of the French National Project. After a preparation phase in 1982-83 (ESOPE project), this project was formally launched in late 1983, with a group of private companies around SG2 as industrial partners.

1.2.4 PROJECT SIZES

As far as the number of people "engaged in the project" is concerned, no easy answer may be given, because of the multiplicity of the experiments. GETA itself has a core of about ten people engaged in software or lingware, plus a varying number of students, not working directly on MT, and some visitors. The core includes two professors and eight research engineers and linguistics (supported by the CNRS), four of them constituting the Russian-French team.

For the National Project, about nine or ten people from GETA or IFCI are working part or full time, while about the same number, coming from the industrial partners, are working full time (as of mid-1984).

2 Application Environment

2.1 PRE-/POST-EDITING

Pre-editing is optional. In our terminology, pre-editing means that some conventional marks are inserted in the input text, which is not otherwise modified, by replacing words with "synonyms" or by rewriting parts to change the syntactical structure. When pre-editing is used, the inserted marks refer to some lexical ambiguity (for example, ambiguity between noun and verb), or indicate the scope of a coordination, the antecedent of a relative pronoun, etc.

The structural analysis grammars are then organized in such a way that these marks, if any, are used first by the disambiguation modules. Until now, this technique has only been employed in the analysis of Portuguese (POR), by P. Daun-Fraga.

Post-editing is necessary in all cases where high quality of the output must be attained, as opposed to situations where information gathering is the main purpose. ARIANE-78 contains a subenvironment for post-editing (REVISION), under which the revisor may simply use the multiwindow editor, or call the THAM subsystem, under which special tools are offered for manipulating and accessing a terminology file.

2.2 HUMAN INTERACTION

During translation, there is normally no human interaction, with the possible exception of morphological analysis, where the operator may correct spelling errors, or insert items in dynamic dictionaries. We don't normally operate this way.

Of course, for debugging purposes, the system may operate in a step-by-step way.

2.3 INTEGRATION OF THE SYSTEM

ARIANE-78 is a complete integrated environment for linguistic programming, debugging, and actual operation.

3 General Translation Approach

The method followed in all current systems written in ARIANE-78 is based on a combination of transfer and interlingua techniques. The process of translation is divided into three phases: analysis, transfer, and generation. The word "synthesis" might be more appropriate, but "generation", as in "code generation" for compilers, is used.

3.2 ANALYSIS

This phase is performed by a sequence of two separate sub-phases: morphological analysis and structural analysis.

3.1.1 MORPHOLOGICAL ANALYSIS

The input is the source text, in ARIANE-78's internal transcription, with optional pre-editing marks, SCRIPT formatting commands, and special occurrences to stand for the *hors-texte* (figures, charts, equations, and other untranslatable material).

The output is a "decorated" (annotated) AND/OR tree structure. The root identifies the text, and its decoration contains ULTXT as value of the UL (lexical unit) "variable" (property, in other terminologies).

The nodes directly under the root correspond to the sentences, and contain the lexical unit ULFRA (FRA for *phrase*, meaning 'sentence' in French).

The nodes at level 2, under the sentence nodes, correspond to the occurrences (running words), and contain the lexical unit ULOCC.

Nodes at level 3 correspond to the result(s) of the morphological analysis. If a given result is "simple", the node contains the corresponding lexical unit, and all grammatical properties, as computed by the grammar. If, on the other hand, a result corresponds to the analysis of the occurrence as a compound word, the node at level 3 contains the lexical unit ULMCP, and dominates a sequence of nodes containing the result of the morphological analysis of the elements of the compound.

The information contained in a solution node (always a leaf) is of the following types:

- the lexical reference, or **lexical unit** – a string;
- grammatical information computed by the grammar from the information on the segments (affixes, root), coming from the dictionaries, such as derivation status, negation status, syntactic category and subcategory, number, tense, person, etc.;
- the syntactic properties of the lexical unit (e.g., the syntactic valencies of a verb or an adjective), which come from the dictionaries;
- the semantic properties of the lexical unit (semantic features and valencies), which also come from the dictionaries.

3.1.2 STRUCTURAL ANALYSIS

The goal of the structural analysis phase is to reach a level of interpretation that is far enough removed from the morphological and syntactic peculiarities of expression in the considered natural language to represent in the same way various expressions having the same "meaning". Furthermore, the interface structure (the result of the structural analysis) keeps a trace of various more "superficial" levels.

To be more concrete, the interface structure of a text is again a decorated tree structure, with the ULTXT and

ULFRA nodes at levels 0 and 1. In a subtree corresponding to a sentence,

- the geometry of the tree structure represents only a bracketing of the sentence in terms of syntagmatic classes (non-terminal categories) such as **verbal clause, noun phrase, adjectival phrase,** etc.
- the relations between words or groups of words are expressed by linguistic attributes (called variables in ARIANE-78's terminology), such as syntactic function (SF), logical relation (RL), or semantic relation (RS).

The syntactic functions may be "subject of a verb", "attribute of the subject", "attribute of the object", "modifier of a clause", etc. The logical relation variable expresses the relations between a predicate (predicative lexical unit such as a verb or certain type of adjective) and its arguments. For example, in the following utterances, the lexical units COMPOUND and CARBON are, respectively, argument 0 and 1 of the predicative lexical unit INCLUDE.

The COMPOUND INCLUDES CARBON...
CARBON is INCLUDED in the COMPOUND
The INCLUSION of CARBON in the COMPOUND

Arguments of predicates are sometimes elsewhere called **inner cases.** When a word or a group of words is related to a predicative unit, without being one of its arguments, we express this relation by some value of the semantic relation property. This is the case for circumstantial complements (sometimes known as **outer cases**). Possible values of RS include cause, condition, consequence, manner, quantification, etc.

Many other properties are contained in the nodes of the structure. Some are only "traces", that is, surface variables such as number or gender. Others are of higher level, for example the determination or actualization properties.

As the units of translation are not restricted to be sentences, but may well include several sentences or paragraphs, it is possible to use the available context to solve anaphora, in case no suitable candidate is found in the sentence. The solution is expressed by copying some of the information relative to the referred element onto the node representing the anaphoric element. This node has the "reference" property set to the adequate value, as well as the "representative" syntactic category.

3.2 TRANSFER

The transfer is also performed in two sequential subphases: lexical transfer and structural transfer.

3.2.1 LEXICAL TRANSFER

During this step, target language lexical units are substituted for the source language lexical units. Several cases must be considered.

- simple-to-simple substitution
- simple-to-complex substitution (a single source unit is translated by several target units, e.g., *avec* → *by means of*)
- complex-to-simple or complex-to-complex substitution (e.g., *computer science* → *informatique, let . . . know* → *informer*).

Moreover, the selection of an appropriate substitution for a given lexical unit may be conditional. The condition may be local, that is, it bears on the properties of the node under consideration (and perhaps some immediate neighbors), or global, in which case a wider context must be examined.

Let us give two examples.

- syntactic valency: the English verb *look* has at least four values for the **object valency** (for argument 1), namely, *at, for, like, after.* According to the syntactic structure that has been built, only one possibility remains after analysis (on the node containing *look*). We may then express the conditional substitution by an item in a TRANSF dictionary of the following (simplified) form:

```
LOOK : if     VAL1=AT    then 'REGARDER'
       elseif VAL1=FOR   then 'CHERCHER'
       elseif VAL1=LIKE  then 'S/OCCUPER'
       else                   'RESSEMBLER'
```

- presence or absence of an argument: the usual translation of *give* is *donner* in French; however, if there is no first argument explicit in the sentence (e.g., *John was given a book*), the translation of *give* may be *recevoir,* with the indication that the third argument of *give* becomes the first argument of *recevoir* (e.g., *Jean a reçu on livre*).

Basically, the TRANSF SLLP provides the means to write bilingual multi-choice dictionaries. Each node of the input tree is replaced by a subtree in the output tree. This subtree may be selected from several possibilities, according to the evaluation of a predicate on the attributes of the input node and its immediate neighbors.

In simple cases, the selected subtree is reduced to one node. In more complex cases, the selected subtree may:

- give several possible equivalents, for further testing in subsequent phases, or production of a multiple equivalent in the final translation (e.g., *process* → *processus* or *procede* from English into French).
- express the prediction that the considered element may be part of a complex expression in the source language (e.g., *let . . . know*). In this case, the subtree will contain nodes describing the type of complex predicted, the other elements of the complex, and the translation of the complex (which may again be simple or complex). It will then be one of the tasks of the structural transfer to confirm or disconfirm this prediction, by using the whole available context, and to take appropriate action; use the translation of the complex if "yes", leave the simple translation if "no".

This organization has been consciously designed in order to limit the cost of indexing in dictionaries: the lexicographers don't have to write complex tree-transformation rules. Instead, they write (static) subtrees of well-defined (sort of and/or) forms, in which some of the information is later used as an indirect call to transformational procedures written by specialized computational linguists in the structural transfer grammar.

3.2.2 STRUCTURAL TRANSFER

We have already begun to explain the role of the structural transfer.

- It must finish a part of the lexical transfer, by using the appropriate context to choose between several remaining equivalents (e.g., on the basis of the semantic features of other elements in the clause), and to handle the translation of complex expressions as explained above.
- If there is no satisfactory universal representation of some phenomenon, the corresponding attributes of the target language are computed in a contrastive way. This is now the case for aspect, tense, and modality of verbs, as well as for determination of noun phrases.
- Finally, it may be necessary to perform true structural transfer operations, although this seldom occurs. In practice, we use this as a "safety net", in cases where the higher levels of interpretation have not been computed by the analysis on some part of the text, so that a mapping between low-level structures is better than nothing.

3.3 GENERATION

Generation too is composed of syntactic followed by morphological generation.

3.3.1 SYNTACTIC GENERATION

Normally, the purpose of syntactic generation is to compute a surface syntactic structure from the interface structures. Starting with the semantic and logical relations, the syntactic functions have to be determined first, and then the syntagmatic classes are computed, in a top-down recursive manner.

The choice between various syntactic structures may be guided by the values of the low-level attributes, as transferred or transformed by the transfer phase. For instance, we may give priority to a passive construction over an active construction, if the (target) interface structure contains the indication of passive. But this passive mark may well have been computed by the transfer (e.g., for an impersonal construction in French preferably translated by a passive in English).

Syntactic transfer will also be used to compute the correct order of the elements in the final translation, as well as all low-level information necessary for morphological generation, such as number, person, tense, gender, etc.

3.3.2 MORPHOLOGICAL GENERATION

The left-to-right sequence of the leaves of the tree resulting from syntactic generation is the input to morphological generation (written in SYGMOR).

The grammar directs the construction of the output occurrence(s) by testing the values of the attributes of the current node (and some other bounded context) and referring to dictionaries of strings, accessed by the UL or other variables, to get the various morphs to be combined (root, prefix, affixes, suffix, . . .).

Also, purely morphological variations are handled by using string-transformation functions (e.g., elision in French, or generation of upper/lower case codes).

4 LINGUISTIC TECHNIQUES

Basically, every linguistic phase is constructed by successively writing two different sets of descriptions.

The first is the analog of the set of specifications for a conventional program. We call it the static description of the desired correspondence, which is usually many-to-many.

B. Vauquois and S. Chappuy have developed a formalism called **static grammars** (Chappuy 1983), not unlike M. Kay's unification grammar formalism but suitable for the kind of structures we are accustomed to manipulate.

A static grammar describes the correspondence between the strings of a natural language and their interface structures. Such a description is neutral with respect to analysis and generation, and does not express any particular strategy for computing the correspondence.

We have yet to devise a similar formalism to describe the correspondence between interface structures of two given natural languages.

The second part of the work consists in writing and debugging the **dynamic (procedural) grammars** in the appropriate SLLPs (ATEF and ROBRA for analysis, ROBRA and SYGMOR for generation). In the most recent applications, the analysis and generation structural grammars are, in effect, (manually) generated from the static grammar of the considered language. Moreover, it is possible to construct several dynamic grammars referring to the same static grammar, in order to experiment with different strategies, trying to make the best of the various possibilities of heuristic programming available in the SLLPs.

4.1 LOW LEVEL

Morphological ambiguities are detected by the morphological analysis phase and handed over to the structural analysis phase. If the analysis is successful, these ambiguities are solved in the process (e.g., noun/verb, etc.). Some polysemic words may be handled as well, if the semantic features associated to the different meanings are distinct enough to be separated by the agreement conditions incorporated in the grammar rules.

4.2 HIGH LEVEL

Although the units of translation are broader than sentences, the bulk of analysis and generation operates essentially at the sentence level. In all current applications, there is no discourse analysis leading to some representation of the "hypersyntax" or "hypersemantics" of the complete unit.

Nevertheless, the possibility to use a wider context is useful in some cases. We have already mentioned anaphoric resolution in analysis. During transfer or generation, it also happens that a sentence of the source text is segmented into several shorter sentences, for stylistic reasons.

All dynamic grammars are written by using production rules (not phrase structure rules) of the various kinds available in the SLLPs. An important feature of ARIANE-78 is that all steps (from morphological analysis to morphological generation) may be ultimately considered as describing transducers, not analyzers.

5 Computational Techniques

5.1 DICTIONARIES

Dictionaries are written in the SLLPs, and then compiled into some internal representation that includes a fast access method. At execution time, the dictionaries reside in virtual memory.

ATEF dictionaries use a two-step hash-coding scheme, followed by an ordered-table representation (for morphs of the same length sharing the same initial or final character). Access to a given item involves less than 100 machine operations.

TRANSF dictionaries use a quasi-perfect hash-coding scheme. SYGMOR dictionary access relies on dichotomy for the lexical units and on sequential search for other variables. But, then, the latter dictionaries are always small and of bounded size anyway: they contain the prefixes, affixes, and endings.

As a matter of fact, due to the compactness of the internal coding and to the speed of the access methods, dictionaries don't raise any computational problem.

5.2 GRAMMARS

In ATEF, the grammar describes a finite-state non-deterministic transducer. It is implicitly divided into as many subgrammars as "morphological formats" (classes). Each item in a dictionary has such a format, which contains some static grammatical information, and is also referenced in the l.h.s. of one of several rules, which will be called in a non-deterministic way.

The underlying mechanism for handling non-determinism is simple backtrack. However, heuristic functions may be called by the rules. Their effect is to "prune" the search space in several predetermined ways. For instance, one of these functions says something like: if the application of the current rule leads to some solution, then don't compute the solutions that might be obtained from segments of strictly shorter length than the current one beginning at the same character. Another heuristic function is used to simply state that, if the rule leads to a solution, this solution should be the only one; previously computed solutions are discarded, and further possibilities are not examined.

In ROBRA, there are several levels of control. First, a given transformational system (TS) has a given store of transformational rules (TR), which operate by substitution. Then there is a collection of grammars (TG), each made of an ordered subset of the TRs. The order is local to the grammar, and is interpreted as a priority order. Each TR may contain a recursive call to a (sub)TG or to a transformational subsystem (sub-TS). The top level of control corresponds to the TS (and its sub-TSs), and is described by means of a **control graph** (CG). The nodes of the CG are the TGs, and a special "exit grammar" (&NUL). Arcs bear conditions of the same type as l.h.s. of rules.

ROBRA's interpreter submits the input tree to the initial grammar of the considered (sub)TS, and uses a built-in back-tracking mechanism to find the first path leading from this initial node to the next node, thereby applying the TGs found in the nodes, and traversing the arcs only if the attached conditions are verified by the current tree. In case of success, the result is the tree that reaches &NUL. In case of failure, the output is set equal to the input.

A "simple" execution of a given TG is carried out in two steps. First, a parallel application of the TRs of the TG is performed, by selecting the maximal (according to some parameters) family of non-overlapping occurrences of rule schemas (l.h.s.) and applying the corresponding rules. Then, the recursive calls, if any, are executed, by submitting the appropriate subtrees to the called sub-TG or sub-TS.

The execution of a TG in " exhaustive" mode consists in iterating simple executions of this TG until no rule is applicable any more. In "controlled" mode, a marking algorithm is used in order to strictly diminish the number of possible occurrences of rules at each iteration, ensuring termination of the process. Hence, it is possible for the compiler to statically detect possible sources of undecidability, just by checking the modes of the TGs, testing for loops in the CG, and verifying a simply condition on the form of the recursive calls.

This kind of organization makes it possible to use text-driven strategies, which will operate differently on different parts of the (tree corresponding to the) unit of translation.

The case of SYGMOR is more simple, because the underlying model is a finite-state deterministic transducer. For each new node (leaf of the input tree), the interpreter selects the first rule whose l.h.s. is verified and executes it. Then, it uses the control part of the rule, which consists of an ordered sequence of rules to be applied, some of which may be optional. Usually, SYGMOR grammars are fairly small.

5.3 EFFICIENCY

The basic software is programmed at various levels. The compilers and interpreters of the SLLPs are written in assembler or PL360, the monitor and the macro for the editor (XEDIT) in EXEC/XEDIT (IBM's VM/CMS's shell language). We can say something about efficiency on two levels.

First, the efficiency of the programming itself. Applications such as Russian-French use roughly 1 to 1.5 Mipw (million instructions per word translated) of VCPU (virtual CPU), measured on a 4321, a 370, a 3081, or an Amdahl. More than 88% of this is used by ROBRA's pattern-matching mechanism, and less than 10% by the "dictionary phases" (morphological analysis and generation, lexical transfer). Translation is performed using 2.5 Mbytes of virtual memory, without any access to secondary storage during processing itself.

Recently, we made a comparison with Kyoto University's system, whose design is similar to ARIANE-78's. In particular, the bulk of the computing time is used by GRADE, a SLLP of the ROBRA family. The system is programmed in UT-LISP and runs on a FACOM computer (Fujitsu), which is IBM-compatible and very fast (20 Mips).

It turns out that this MT system uses roughly 100 times more Mipw than ARIANE-78, and 40 times more space. Taking into account the fact that there is only 4 Mbytes of virtual memory (divided by 2 for the purpose of garbage collection), that the garbage collector gobbles up 40% of the VCPU, and that the runtime access to the 30 to 40 Mbytes of secondary storage (holding the lingware) takes 20% more, we end up with the net result that *a LISP implementation* (of this type of system) *is 40 times more voracious in computer time and space than a low-level implementation.*

Of course, the amount of programming and maintenance effort is higher for the latter type of implementation. At this point, it is worth remembering that in France, contrary to many countries, research labs usually have access to severely limited computer resources, and must pay for it. Natural Language Processing is very much an experimental science, and the designers of ARIANE-78 have felt they couldn't provide the linguists with a system they might use for experimentations only about one or two weeks a year because of financial constraints.

One possible reward of this painful kind of implementation is that it seems possible to run the complete system on the PC/XT370, according to the specifications and descriptions published in Europe.

A second level of comparison is by the computational methods used. For that, we may use old data from the former CETA system (before 1971), or data from current systems such as METAL (University of Texas at Austin), or KATE (KDD, Tokyo), based on augmented context-free formalisms. From some demonstrations and private communications, we got again the figure of 1 to 1.5 Mipw, for systems written in LISP, and about 40 times less for assembler-level implementation of the basic software.

Our (perhaps too hasty) conclusion is that *pattern-matching-based techniques are roughly 40 times costlier in VCPU than classical combinatorial methods* (if programmed in a smart way, of course).

However, we have tried the latter kind of approach in the past, and ended up finding it quite difficult to maintain a large scale system and to raise its overall quality. See Section 7 for further comments.

6 Practical Experience; Experimental Results

6.1 COST OF SYSTEM MAINTENANCE AND EXTENSION

This cost is quite low, in terms of compute resources and of human effort. The compilers are quite efficient, the structure of dictionary and annex tools is designed to simplify indexing and correcting, and the modular character of the grammars makes it quite easy to modify and debug them.

6.2 HOW IS TESTING PERFORMED?

Take the Russian-French application as an example. For testing, we use a set of corpora we retranslate each time a significant set of modifications has been incorporated in the lingware, and of course we translate new corpora that have just been inputted and checked for spelling errors. We then compare old and new translations of the first set, and revise the second.

6.3 WHAT IS MEASURED?

6.3.1 SPEED

As mentioned earlier, we measure the VCPU time and deduce from it an estimate in terms of Mipw, which has proven to be quite stable over a variety of machines. Due to the differences in real memory size and processing speed, the real time (even in a mono-user situation) may lay anywhere from 100 words per hour to 5 words per second.

6.3.2 QUALITY

This is a difficult thing to measure. During revision, we do such a qualitative evaluation, by detecting all translation errors and trying to find their origin, hence their gravity.

However, we would prefer to simply measure the time for human revision (without error hunting), as compared with the time for human revision of an average human rough translation of the same text. This kind of experiment has yet to be performed in a meaningful way.

6.4 COST EFFECTIVENESS, SUBJECTIVE FACTORS

This has to be measured in some real situation, as said earlier. We strongly hope that the setting of the National Project will allow that kind of measurement to take place in a systematic way.

7 ARGUMENTS RE CHOSEN TECHNIQUES

7.1 AGAINST ALTERNATIVE METHODS

7.1.1 INTERLINGUA

First of all, we have tried an approximation of the interlingua ("pivot") approach, and found it wanting. In the former CETA system, the pivot representation was of a hybrid sort, using as vocabulary the lexical units of a given natural language, and as relations so-called "universals" corresponding to our current logical and semantic relations, plus abstract features such as semantic markers, abstract time and aspect, and so on. The problem here is threefold.

- It is very difficult to design such a pivot in the first place, and ever more so if the vocabulary must also be independent of any particular natural language.
- The absence of surface-level information makes it impossible to use contrastive knowledge of two languages to guide the choice between several possible paraphrases at generation time.
- If the high-level representation cannot be computed on part of the unit of translation, the whole unit gets no representation, and hence no translation, or a word-by-word translation. This is already bad enough at the sentence level, and quite insufferable if the units are larger, in order to access a bigger context.

7.1.2 PS-RULES

Phrase structure (PS) grammars don't seem adequate for our purposes, even with all (not so recent) additions and niceties such as attributes, validation/invalidation between rules, attached transformations, etc. This is mainly so because the structures associated with the strings are grammar-dependent, although they should be language-dependent invariants.

Another problem comes from the monolithic aspect of such grammars, which makes them very difficult and ultimately impossible to understand and modify, although everything seems right at the beginning, with a few hundred rules. Stratification of the grammar in the METAL sense is just a device allowing conservation of results obtained in simple cases while rules are added to take care of more complex situations. For the latter, the grammar is just the collection of all rules, with no modularity.

7.1.3 PURELY COMBINATORIAL METHODS

Ambiguity is a fundamental problem in Natural Language Processing. Combinatorial methods tend to compute all possibilities, perhaps weighting them, and to filter out the first one, the first N ones, or all of them. No heuristic programming is possible.

If more than one result of analysis (or transfer) is produced, the source of the ambiguity is lost, so that the system must produce several distinct translations for the unit. Again, this is difficult to accept at the sentence level, and certainly unacceptable at the paragraph level.

7.2 FOR CHOSEN METHODS

7.2.1 CONSTRASTIVITY AND TRANSFER APPROACH

If we are some day to attain the level of performance of an average or good translator, it is unavoidable to sometimes rely on "rules of thumb", which use surface-level information, and embody some contrastive knowledge of the languages at hand.

Moreover, if the units of translation grow larger, the probability that some part cannot be completely analyzed at the most abstract levels of interpretation approaches certainty.

Hence, we feel that the use of one (and not several) so-called "multilevel structure" for representing each unit is appropriate. As a matter of fact, we consider such a structure as a generator of the various structures that have been implicitly computed at each level of interpretation.

This technique may be compared with the "blackboard" technique of some AI systems. During analysis, the different levels of linguistic knowledge are used in a cooperative way, and not sequentially, as in previous systems.

For example, some semantic information may be used to disambiguate at the syntactic level, as in the following sentences:

John drank a bottle of beer.
John broke a bottle of beer.

7.2.2 TRANSDUCERS RATHER THAN ANALYZERS

Although procedural programming is notoriously more difficult than writing a collection of static rules used by a standard algorithm, leading to a yes or no answer, we consider it a lot better in the situation of incomplete and fuzzy knowledge encountered in MT. It happens that the same position has recently gained ground in AI, with the construction of "expert systems" that embody a lot of knowledge in their control and domain-specific heuristics.

In our case, this amounts to using our SLLPs for "language engineering", in much the same way as usual programming languages are used for software engineering. Starting from a kind of functional specification (expressed by means of static grammars), the computational linguist constructs a corresponding transducer in the time-honoured way of top-down decomposition and step-wise refinement.

7.2.3 TEXT-DRIVEN HEURISTIC PROGRAMMING

Heuristic programming and text-driven strategies seem more adequate than the use of a very complex grammar, whose rules are all tried, even in simple cases. Experiments have show that the flow of control (from TG to TG) is significantly different on different parts (subtrees) of the translation units.

7.2.4 FAIL-SOFT MECHANISMS

In the setting of second-generation systems, based on implicit rather than explicit understanding, a parallel can be made with compilers for programming languages. We don't want our "supercompilers" (for natural language, that is) to stop and produce nothing if they encounter an ill-formed clause somewhere in the unit of translation.

Rather, we want them to produce the best translation they can, under all circumstances, annotating it with special marks, analogous to error and warning messages, to be used later during the post-editing and technical revision of the document.

The "safety net" made of the multiplicity of levels of interpretation available on the same structure makes it possible to use a broad spectrum between high-level and word-by-word translation.

8 Future Directions

8.1 APPLICATION PLANS

Immediate application plans are all essentially those of the French National Project. As first target, some kind of aircraft manuals have been selected for translation from French into English. Computer manuals to be translated from English into French may follow.

We also hope to develop current cooperative efforts and initiate new ones.

8.2 RESEARCH PLANS

Lexical knowledge processing is one of the most important research topics, from a practical point of view. We aim at designing a kind of "integrated dictionary" where the "coded" and "natural" aspects are mixed. Such dictionaries could be used as reference or interface structures, e.g., to generate coded dictionaries in different SLLPs for various steps of MT, or for other tasks, such as spelling correction, etc.

Grammatical knowledge processing is being investigated. The formalism of static grammars is being refined and experimented with for the string-to-tree correspondences, and a small team is currently designing a set of software tools to handle a data base of static grammars. The formalism should also be extended to tree-to-tree correspondences. Many useful ideas can be taken over from current work on specification, proof, and validation of programming languages.

A promising topic is the introduction of expert systems components into second-generation MT systems in order to add some level of "explicit understanding" in the form of domain-specific "extralinguistic knowledge" and typology-specific "metalinguistic knowledge".

Research on better data- and control-structures for adequate SLLPs has been carried out. It should be continued and experimented with.

Acknowledgments

We thank the referee for helping us remove some linguistic bugs from this paper, and for indicating some points that needed clarification. We hope we have provided a better explanation of these.

References

Bachut, D. and Verastegui, N. 1984 Software Tools for the Environment of a Computer-Aided Translation System. *Proceedings of COLING-84*. Stanford, California.

Boitet, C. 1976 Un essai de réponse à quelques questions théoriques et pratiques liées à la Traduction Automatique. Définition d'un système prototype. Thèse d'Etat. Grenoble.

Boitet, C. 1984 Research and Development on MT and Related Techniques at Grenoble University (GETA). Lugano Tutorial on Machine Translation.

Boitet, C. and Gerber, R. 1984 Expert Systems and Other New Techniques in MT. *Proceedings of COLING-84*. Stanford, California.

Boitet, C.; Guillaume, P.; and Quézel-Ambrunaz, M. 1978 Manipulation d'arborescences et parallélisme: le système ROBRA. *Proceedings of COLING-78*. Bergen.

Boitet, C.; Guillaume, P.; and Quézel-Ambrunaz, M. 1982 ARIANE-78: an Integrated Environment for Automated Translation and Human Revision. *Proceedings of COLING-82*.

Boitet, C. and Nedobejkine, N. 1981 Recent Developments in Russian-French Machine Translation at Grenoble. *Linguistics* 19: 199-271.

Chappuy, S. 1983 Formalisation de la description des niveaux d'interprétation des langues naturelles. Theèse de 3è cycle. Grenoble.

Chauché, J. 1974 Transducteurs et arborescences. Etude et rèalisation de systèmes appliqués aux grammaires transformationnelles. Thèse d'Etat. Grenoble.

Clemente-Salazar, M. 1982 Etudes et algorithmes liées à une nouvelle structure de données en TA: les E-graphes. Thèse de Docteur-Ingénieur. USMG and INPG, Grenoble. See also *Proceedings of COLING-82*. Prague.

Gerber, R. 1984 Etude des possiblités de coopëration entre un système fondé sur des techniques de comprëhension implicite (système logico-syntaxique) et un système fondé sur des techniques de compréhension explicite (système expert). Thèse de 3è cycle. Grenoble.

Mozota, T. 1984 Un formalisme d'expressions pour la spécification du contrôle dans les systèmes de production. Thèse de 3è cycle. Grenoble.

Vauquois, B. 1975 La Traduction Automatique à Grenoble. Document de Linguistique Quantitative 29. Dunod, Paris.

Vauquois, B. 1979 Aspects of Automatic Translation in 1979. Scientific Program, IBM, Japan.

Vauquois, B. 1983 Automatic Translation. *Proceedings of the Summer School "The Computer and the Arabic Language."* Rabat: Chapter 9.

Verastegui, N. 1982 Etude du parallélisme appliquë à la traduction automatisée par ordinateur. STAR-PALE: un système parallèle. Thèse de Docteur Ingénieur. USMG and INPG, Grenoble. See also *Proceedings of COLING-82*. Prague.